Estágio Científico e Tecnológico I - EE015

Relatório

Predição de Séries Temporais Baseada em Redes Neurais Artificiais

Submetido à
Faculdade de Engenharia Elétrica e Computação (FEEC)

Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (UNICAMP)
CEP 13083-852, Campinas - SP

Aluno: João Pedro de Oliveira Pagnan
Orientador: Prof. Levy Boccato

Campinas, 7 de julho de 2021

1 Introdução

A predição de séries temporais é uma das aplicações mais interessantes do tratamento de informação. O desafio de antecipar padrões de comportamento e construir modelos que sejam apropriados para explicar determinados fenômenos da natureza tem importância para a biologia, economia, automação industrial, meteorologia e diversas outras áreas da ciência [1].

É possível definir uma série temporal como sendo um conjunto de observações de uma grandeza ou fenômeno de interesse tomadas durante um intervalo de tempo. Os sistemas cujas medições formam uma série temporal podem ser originados por processos determinísticos ou estocásticos [1].

Através da análise e interpretação de uma série temporal, podemos estimar os seus valores futuros, aumentando a informação que podemos obter das observações que já foram realizadas em um sistema.

Na literatura, encontramos diversos tipos de modelos para a predição de séries temporais, desde métodos clássicos lineares, como o modelo autorregressivo (AR) [1] até métodos não-lineares utilizando, por exemplo, redes neurais artificiais, sendo que dessas se destacam as redes do tipo *Multilayer Perceptron* (MLP) e as redes recorrentes, especialmente a *Long Short-Term Memory* (LSTM) [2] e a *Echo State Network* (ESN) [3].

Uma classe de sistemas dinâmicos particularmente relevante dentro do contexto de modelagem e predição de séries temporais está ligada à ideia de dinâmica caótica. Diversos fenômenos naturais, como a dinâmica populacional de uma espécie, a dinâmica atmosférica de uma região, ou até mesmo as órbitas de um sistema com três ou mais corpos celestes podem exibir comportamento caótico. Apesar de serem determinísticos (e, portanto, previsíveis), esses sistemas são extremamente sensíveis às condições iniciais [4]. Isso causa um problema para a predição das séries temporais originadas por eles, pois uma pequena incerteza na medida afetará toda a previsão.

Tendo em vista o desempenho de modelos não-lineares para previsão de diversas séries temporais [2], optamos por estudar a aplicabilidade de redes neurais artificiais à previsão de séries relacionadas a sistemas com dinâmica caótica.

Essa primeira parte do projeto de iniciação científica teve como objetivo estudar a base teórica das redes neurais artificiais e de outros regressores lineares clássicos, assim como estudar os fundamentos de sistemas dinâmicos e de dinâmica caótica.

O estudo dirigido começou abordando uma revisão de tópicos de probabilidade, teoria da informação e estimação. Em seguida, foi vista a teoria de regressores e classificadores clássicos [5]. Depois disso, o estudo se dirigiu para as redes neurais artificiais MLP e recorrentes [6], para, por fim, concluir o aprendizado de preditores com uma breve exposição dos modelos autorregressivos (AR) e autorregressivos de médias móveis (ARMA) [1]. Com a teoria de predição solidificada, o foco mudou para os fundamentos da teoria de sistemas com dinâmica caótica, utilizando como base as referências [4] e [7].

Iniciamos nossa discussão pela Seção 2, na qual são apresentados alguns conceitos básicos de sistemas dinâmicos, seguidos de uma exposição sobre sistemas caóticos, evidenciando as particularidades que eles têm em relação aos sistemas dinâmicos convencionais.

Em seguida, na Seção 3, veremos os modelos de predição baseados em redes neurais artificiais, juntamente com uma breve exposição de modelos lineares básicos.

Por fim, na Seção 4 são indicados os próximos passos deste projeto de iniciação científica visando sua conclusão no final do segundo semestre de 2021.

2 Sistemas Dinâmicos

O estudo de sistemas dinâmicos se tornou fundamental para a ciência quando a humanidade começou a desenvolver modelos matemáticos para explicar o mundo em sua volta, utilizando os modelos criados para realizar predições e, de certa forma, saber o que nos espera no futuro.

A teoria de sistemas dinâmicos se mostra útil para a modelagem e análise de diversos fenômenos físicos, biológicos, químicos, ou equações matemáticas, através do tempo. Nesse caso, o tempo pode ser tanto contínuo, quanto discreto [8].

Os sistemas dinâmicos também podem ser lineares ou não-lineares. Um sistema linear, sendo este a tempo contínuo ou discreto, possui a propriedade da superposição, ou seja, se uma entrada consiste de uma soma ponderada de diversos sinais, então a saída é a superposição das respostas do sistema a cada um desses sinais [9].

É possível representar essa propriedade através das seguintes relações matemáticas, sendo $y(t)$ a resposta de um sistema a tempo contínuo a uma entrada $x(t)$ e $y_i(t)$ a saída correspondente a $x_i(t)$:

$$\text{A resposta a } x_1(t) + x_2(t) \text{ é } y_1(t) + y_2(t) \tag{1a}$$

$$\text{A resposta a } a \cdot x_1(t) \text{ é } a \cdot y_1(t), \text{ em que } a \text{ é qualquer constante complexa.} \tag{1b}$$

A primeira equação (1a) representa a propriedade da aditividade, enquanto a segunda (1b) representa a propriedade da homogeneidade. Essa definição também é válida para sistemas a tempo discreto, e podem ser combinadas:

$$a \cdot x_1(t) + b \cdot x_2(t) \rightarrow a \cdot y_1(t) + b \cdot y_2(t) \tag{2a}$$

$$a \cdot x_1[n] + b \cdot x_2[n] \rightarrow a \cdot y_1[n] + b \cdot y_2[n], \tag{2b}$$

sendo a e b constantes complexas quaisquer [9].

Devido a todo arcabouço teórico de álgebra linear elaborado por matemáticos nos últimos séculos, trabalhar com modelos lineares de sistemas dinâmicos se tornou algo bem estabelecido. Logo, grande parte dos cursos de graduação lidam, predominan-

temente, com sistemas lineares por definição, ou aplicam métodos de linearização em sistemas não-lineares, chegando em soluções analíticas lineares aproximadas, com um bom grau de acurácia sob certas condições de contorno [4].

Com essas aproximações, prever o valor futuro de uma série temporal originada por um sistema linear (ou um sistema não-linear linearizado) é algo relativamente simples, afinal, pode-se utilizar a solução analítica.

Porém, a esmagadora maioria dos fenômenos que observamos na natureza é altamente não-linear. Dessa forma, ou não conseguimos aplicar os métodos de linearização que tanto aprendemos ou, ao serem aplicados, o sistema se reduz a uma representação bem imprecisa da sua dinâmica original [10].

Antes de falarmos sobre a teoria da dinâmica caótica, que trouxe diversas explicações para sistemas não-lineares e aumentou o entendimento da ciência sobre esses fenômenos, veremos um pouco sobre alguns conceitos fundamentais no estudo de sistemas dinâmicos.

2.1 Espaço de Fase

Algo comum no estudo de sistemas dinâmicos é a representação em espaço de fase (também chamado de espaço de estado). Os eixos do espaço de fase representam os graus de liberdade de um sistema dinâmico. Assim, cada ponto no espaço de fase representa um estado possível para o sistema.

Por exemplo, o movimento de um pêndulo simples, sendo θ o ângulo entre o pêndulo e o eixo vertical, g a aceleração gravitacional, e l o comprimento do pêndulo, é descrito pela seguinte equação diferencial:

$$ml^2 \cdot \ddot{\theta}(t) + mgl \cdot \operatorname{sen}(\theta(t)) = 0 \tag{3}$$

Apesar de não ser possível obter uma solução analítica sem antes realizar algum tipo de linearização, é possível estudar a evolução temporal desse sistema através do seu espaço de fases [4].

Para tal, escreveremos (3) da seguinte forma:

$$\dot{\theta} = \varphi = f(\theta, \varphi) \tag{4a}$$

$$\dot{\varphi} = -\frac{g}{l} \cdot \operatorname{sen}(\theta) = g(\theta, \varphi) \tag{4b}$$

Derivando a equação (4b) em θ e isolando os termos θ e φ em lados opostos:

$$\varphi \, d\varphi = -\frac{g}{l} \cdot \operatorname{sen}(\theta) \, d\theta \tag{5}$$

Agora, integrando (5) em ambos os lados, chega-se a:

$$\varphi^2 - \frac{2g}{l} \cdot \cos(\theta) = C \tag{6}$$

sendo C uma constante arbitrária. A Figura 1 mostra a trajetória no espaço de fases para diversos valores de C (lembrando que $\varphi = \dot{\theta}$).

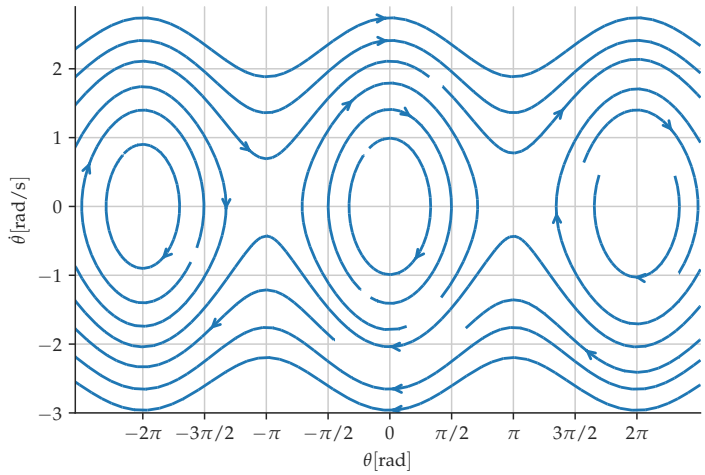

Figura 1: Trajetória no espaço de fase do pêndulo simples para vários valores de C

Perceba que um ponto $(\theta, \dot{\theta})$ no espaço de fases representa um estado possível para o sistema. Nesse caso, como não há nenhum elemento estocástico na evolução do pêndulo simples (considerando apenas a mecânica clássica), não há intersecções nas trajetórias no espaço de fases.

Por fim, é pertinente dizer que o espaço de fases pode ter quantas dimensões forem necessárias para representar a dinâmica de um sistema.

2.2 Atratores

Uma estrutura topológica utilizada para estudar a evolução de um sistema dinâmico, a qual é capaz de capturar a essência de sua evolução temporal, é o atrator.

De acordo com a definição dada em [4], um atrator é um conjunto invariante para o qual órbitas próximas no espaço de fase convergem depois de um tempo suficientemente grande.

Em termos matemáticos [4]: Uma região compacta A é um atrator de um fluxo $\varphi(t, x)$ se as quatros hipóteses a seguir valem:

a) A é invariante segundo φ;

b) A tem uma vizinhança contraente;

c) o fluxo é recorrente, isto é, trajetórias começando em qualquer subconjunto aberto de A voltam a esse subconjunto para valores de t suficientemente longos;

d) o fluxo não pode ser decomposto, isto é, A não pode ser dividido em duas partes invariantes não triviais.

O formato do atrator varia bastante de acordo com o sistema em análise. Por exemplo, um circuito RLC autônomo tem um atrator chamado de ponto-fixo. Para evidenciarmos isso, considere que as seguintes equações diferenciais descrevem a dinâmica de um circuito RLC, sendo i a corrente no indutor e v a tensão no capacitor [7]:

$$\frac{di}{dt} = -\frac{1}{L} \cdot (R \cdot i(t) + v(t)) \tag{7a}$$

$$\frac{dv}{dt} = \frac{i(t)}{C} \tag{7b}$$

Assim, dada uma energia inicial presente no sistema, seja esta em forma de campo elétrico no capacitor, ou campo magnético no indutor, o resistor dissipará a energia armazenada em forma de potência ativa (calor). Considerando as variáveis do espaço de estado como sendo a tensão no capacitor e a corrente do indutor, é fácil perceber que a trajetória vai convergir para o ponto $(0, 0)$, pois a energia será toda dissipada e não haverá mais dinâmica nesse sistema.

A Figura 2 mostra a evolução de um circuito RLC em série para $v(0) = 1$, $i(0) = 1$ com $R = 1$, $L = 1000$ e $C = 500$:

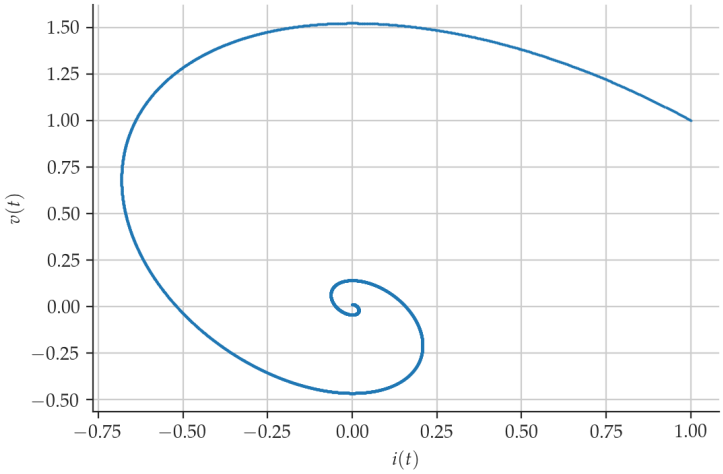

Figura 2: Representação do atrator ponto-fixo presente no circuito RLC série

Os atratores podem ser pontos-fixos, ciclo-limites (que é o caso do oscilador harmônico e do pêndulo simples para certos valores iniciais de θ), pontos de nó, pontos de sela, entre outros. Os atratores também podem indicar trajetórias instáveis, estáveis e assintoticamente estáveis. Também é válido mencionar que a região do espaço de fase para a qual todas as trajetórias presentes seguem um determinado atrator representa a bacia de atração daquele atrator.

O tipo de atrator e o tipo de estabilidade é obtido analisando os autovalores λ_i da representação matricial do sistema dinâmico.

Por fim, quando há uma mudança no sistema dinâmico que faz com que ele seja atraído para a bacia de outro atrator, dizemos que o sistema sofreu uma bifurcação. Geralmente são os parâmetros do sistema que definem os pontos de bifurcação [4].

2.3 Expoentes de Lyapunov

Considere um sistema contínuo com m equações diferenciais ordinárias, e imagine um pequeno hiper-volume esférico de teste de estados iniciais vizinhos y_0 (raio $\varepsilon_0(x_0)$) em torno do ponto inicial x_0 de uma linha de fluxo, isto é [4]:

$$|y_0 - x_0| \leq \varepsilon_0(x_0) \tag{8}$$

Com o passar do tempo, o fluxo das trajetórias deforma a hiper-esfera num objeto hiper-elipsoidal com eixos principais $\varepsilon_k(t)$, $k = 1, 2, ..., m$ (Figura 3). Os expoentes de Lyapunov medem o crescimento exponencial dos eixos principais $\varepsilon_k(t)$ e são definidos por:

$$\lambda_i = \lim_{t\to\infty} \lim_{\varepsilon_0(x_0)\to 0} \frac{1}{t} \ln \frac{\varepsilon_i(t)}{\varepsilon_0(x_0)}, \quad i = 1, ..., m. \tag{9}$$

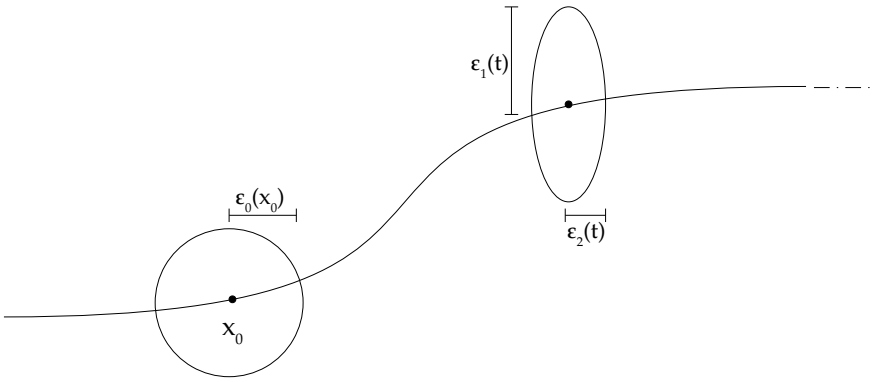

Figura 3: Evolução de um elemento de volume esférico de raio $\varepsilon_0(x_0)$ em torno de um ponto inicial x_0

A definição em (9) permite a seguinte aproximação:

$$\varepsilon_i(t) \approx \varepsilon_0(x_0) \cdot e^{\lambda_i t} \tag{10}$$

Da equação (10), conclui-se que [4]:

a) a existência de um ou mais expoentes de Lyapunov positivos define uma instabilidade orbital nas direções associadas;

b) uma solução caótica tem pelo menos um expoente de Lyapunov positivo (falaremos mais sobre esse fato adiante);

c) uma solução periódica ou quasi-periódica possui $\lambda_i < 0$ nas direções perpendiculares ao movimento no espaço de fase, e $\lambda_i = 0$ ao longo da trajetória.

Agora, considerando que um elemento de hiper-volume $\delta V(t)$ em um instante t é dado por

$$\delta V(t) = \prod_{i=1}^{m} \varepsilon_i(t), \tag{11}$$

tem-se que, substituindo (11) em (10):

$$\delta V(t) = \delta V(0) \cdot \exp\big(\sum_{i=1}^{m} \lambda_i t\big) \tag{12}$$

A partir dessa definição, é possível concluir que o hiper-volume no espaço de fase não diverge : a) quando $\sum_{i=1}^{m} \lambda_i = 0$, ou seja, quando $\delta V(t) = \delta V(0)$, o que determina um sistema conservativo; b) quando $\sum_{i=1}^{m} \lambda_i < 0$, ou seja, quando $\delta V(t) < \delta V(0)$, o que determina um sistema dissipativo [4].

Por fim, é possível identificar o atrator do sistema dinâmico em análise através do sinal dos expoentes de Lyapunov. Por exemplo, no caso tridimensional ($m = 3$), caso todos os expoentes λ_i sejam negativos, o atrator será um ponto fixo. Já para dois expoentes negativos e um nulo, têm-se um atrator do tipo ciclo-limite.

Os atratores de um sistema caótico possuem uma propriedade especial com relação a seus expoentes de Lyapunov. Esse fato será discutido na Seção 2.5.2.

2.4 Entropia de Kolmogorov-Sinai

Também é possível analisar a dinâmica de um sistema através de uma abordagem via teoria da informação.

Para tal, considere um atrator no espaço de estados, recoberto por cubos de lado ε. Agora, seja uma sequência de b cubos $i_1, i_2, ..., i_b$, e seja $p(i_1, i_2, ..., i_b)$ a probabilidade que o sistema se encontre sucessivamente nos cubos $i_1, i_2, ..., i_b$ em intervalos de tempo regulares τ [4, 7]. Segundo Shannon [11], a informação associada à sequência b é proporcional a:

$$I_b^{(1)}(\varepsilon) = \sum_{i_1, i_2, ..., i_b} p(i_1, i_2, ..., i_b) \cdot \ln p(i_1, i_2, ..., i_b) \tag{13}$$

Perceba que a grandeza:

$$I_{b+1}^{(1)}(\varepsilon) - I_b^{(1)}(\varepsilon) \tag{14}$$

determina a informação necessária para se localizar o estado do sistema (determinar

o cubo i_{b+1}) sabendo que ele percorreu a sequência de cubos $i_1, i_2, ..., i_b$. A grandeza expressa por (14) quantifica a perda de informação do observador (e o ganho de informação do sistema, consequentemente) entre os instantes de tempo $b\tau$ e $(b+1)\tau$ [7].

Com isso, define-se a entropia de Kolmogorov-Sinai como a taxa média de criação de informação no sistema (ou perda de informação do observador) [4, 7]:

$$K = \lim_{\tau \to 0} \lim_{\varepsilon \to 0} \lim_{b \to \infty} \frac{1}{b\tau} \sum_{i_1, i_2, ..., i_b} p(i_1, i_2, ..., i_b) \cdot \ln p(i_1, i_2, ..., i_b) \tag{15}$$

A análise da entropia de Kolmogorov-Sinai de um sistema dinâmico revela várias propriedades do comportamento desse sistema ao longo do tempo. Por exemplo, analisando um sistema cuja trajetória segue a bacia de atração de um ponto-fixo, necessitamos de pouca informação para predizer com precisão qual será o estado do sistema em um instante de tempo posterior. Ou seja, não há produção significativa de informação por parte do sistema. Isso também é válido para um sistema cujo atrator é um ciclo-limite.

A entropia de Kolmogorov-Sinai possui uma conexão com os expoentes de Lyapunov. No caso, [7, 4] mostram que as seguintes equações são válidas, sendo λ_i os expoentes de Lyapunov:

$$\frac{dI_i^{(1)}}{dt} \propto \lambda_i \tag{16a}$$

$$K \leq \sum_{\lambda_i > 0} \lambda_i \tag{16b}$$

De acordo com a relações em (16), podemos concluir que, num sistema determinístico regular (linear e não-linear sem ser caótico ou estocástico), pontos próximos no espaço de fase evoluem sem divergências significativas. Logo, o sistema não produz informação. Assim, a entropia de Kolmogorov-Sinai será nula.

Interessantemente, é visto que, para sistemas estocásticos, a evolução da trajetória no espaço de estado não seguirá nenhum tipo de continuidade, havendo uma perda abrupta de informação por parte do observador. Logo, $K \to \infty$ [7].

Por outro lado, sistemas com dinâmica caótica, apesar de terem uma produção ininterrupta de informação (Seção 2.5.3), têm entropias que seguem a relação dada em (16b).

Portanto, para vermos mais detalhes do porquê disso ocorrer, na seção a seguir iniciaremos nossa discussão sobre sistemas caóticos.

2.5 Sistemas com Dinâmica Caótica

Uma classe de sistemas não-lineares importante para o estudo de diversos fenômenos é a classe de sistemas caóticos. Apesar de ter tido certa dificuldade para encontrar um meio dentro da academia e ser bastante renegada por vários cientistas da época por, entre outros fatores, inicialmente basear-se em simulações computacionais de dinâmicas físicas, algo que não era comum em meados do século XX [10], a teoria da dinâmica caótica foi adotada por cientistas da área da física, matemática, química, biologia, economia, astrofísica, e várias outras, trazendo explicações para diversos fenômenos que não eram bem entendidos até então.

Discutiremos agora algumas propriedades dos sistemas caóticos que os diferem de sistemas lineares e não-lineares convencionais.

2.5.1 Dimensões fractais no espaço de fase de uma dinâmica caótica

Como foi dito anteriormente, as trajetórias no espaço de fase de um pêndulo simples não se interseccionam. Isso é devido ao fato de que, como o sistema é determinístico, se ele estiver em um estado $(\theta(t), \dot{\theta}(t))$, só existe um próximo estado possível $(\theta(t + dt), \dot{\theta}(t + dt))$, considerando que é um sistema isolado, sem forças externas. Caso houvesse a intersecção de trajetórias, o sistema teria dois ou mais próximos estados possíveis, o que não é coerente com um sistema determinístico [4].

Apesar de apresentar um comportamento errático e, de certa forma, imprevisível, sistemas caóticos são determinísticos. Ou seja, também não há intersecções em suas trajetórias no espaço de fase. Mas, como esses sistemas são capazes de produzir um comportamento tão variado, sem apresentar nenhum tipo de ordem aparente, sem haver intersecções nas trajetórias? Para explicar isso, precisaremos generalizar a noção de dimensão para englobar as dimensões fractais [12].

O estudo de estruturas topológicas com dimensão fractal tornou-se relevante na matemática nos anos 70 [10]. Utilizando esse conceito, temos que, além das dimensões inteiras que aprendemos a lidar durante nossa formação (quadrados, cubos, hipercubos etc.), as dimensões fractais representam valores fracionários para essa medida.

Como a base matemática de álgebra linear não engloba essas figuras (pelo menos não no conteúdo que geralmente aprendemos durante a graduação), é necessário o uso de outras ferramentas para estimar a dimensão fracionária. Uma delas é a dimensão de Hausdorff, ou dimensão de Hausdorff-Besicovitch [7, 4].

Seja um conjunto de pontos A num espaço de dimensão N e sejam estes pontos recobertos por hiper-cubos de aresta ε. A dimensão de Hausdorff é dada por:

$$D_0 = \lim_{\varepsilon \to 0} \frac{\log K(\varepsilon)}{\log(1/\varepsilon)} \tag{17}$$

Essa definição é válida tanto para figuras de dimensão inteira, como quadrados e

cubos, quanto para objetos de dimensão fractal.

Aplicando essa definição numa trajetória no espaço de fase de uma dinâmica caótica, vê-se que a dimensão obtida sempre será fracionária [4]. Logo, pode-se dizer que a trajetória sofrerá sucessivas contrações em certas direções, e expansões em pelo menos uma, de forma a nunca se cruzar, por mais errático que seja o movimento [4].

Uma das propriedades importantes de objetos topológicos de dimensão fractal é que eles apresentam uma auto-similaridade [10]. Nesse caso, partes da trajetória no espaço de fase têm uma estrutura análoga à da trajetória como um todo [7]. Ou seja, ao ampliar-se um objeto fractal, suas características serão similares ao do objeto como um todo, tendo uma auto-similaridade.

A Figura 4 evidencia essa propriedade para o mapa de Hénon [13], que é definido pelas seguintes equações a diferenças:

$$x[n+1] = y[n] + 1 - a \cdot (x[n])^2 \tag{18a}$$

$$y[n+1] = b \cdot x[n] \tag{18b}$$

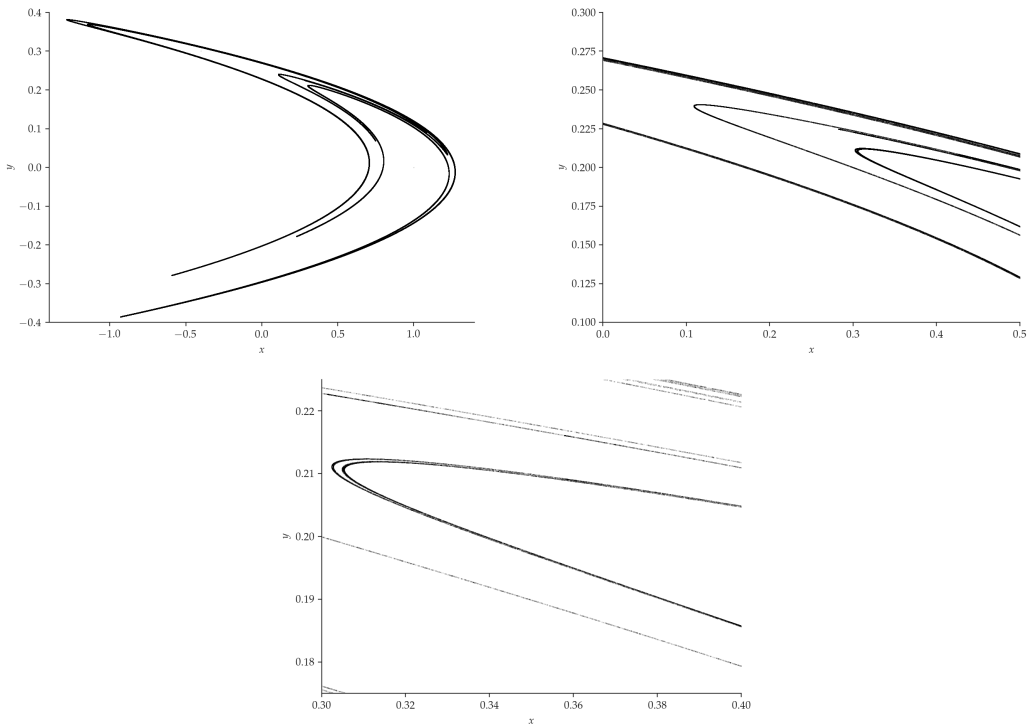

Figura 4: Auto-similaridade presente no mapa de Hénon. Observe o quão similares são as formas observadas nas três figuras, as quais foram obtidas após sucessivas ampliações de trechos específicos nas variáveis x e y no espaço de fase.

Devido à dimensão fractal e a outros fatores, denominamos os atratores de sistemas caóticos de atratores estranhos [14], sendo a própria estrutura presente no mapa de Hénon um atrator estranho. Veremos mais sobre eles na próxima seção.

2.5.2 Atratores Estranhos

O termo atrator estranho foi inicialmente cunhado pelos matemáticos David Ruelle e Floris Takens no artigo [14], para descrever o fluxo em um fluido turbulento. Esse nome foi dado devido às estranhas propriedades e características dessas estruturas topológicas que as diferenciavam dos atratores convencionais que foram citados na Seção 2.2 [10].

De acordo com [7], sendo A um conjunto limitado num espaço de N dimensões, dizemos que A é um atrator estranho se for um atrator (utilizando as definições da Seção 2.2) e se, estando a condição inicial x_0 num subconjunto de pontos próximos, as evoluções temporais desses pontos apresentarem dependência sensitiva em relação às condições iniciais (DSCI) [4].

O ponto que torna esses atratores tão distintos dos usuais é que, no caso linear, espera-se que quando as condições iniciais são próximas, a evolução temporal do sistema no espaço de fase será bem semelhante. A Figura 5 evidencia esse fato para o caso estudado na Seção 2.2, do circuito RLC, onde a diferença entre os estados $(i(t), v(t))$ para as condições iniciais testadas é reduzida com o avanço da simulação.

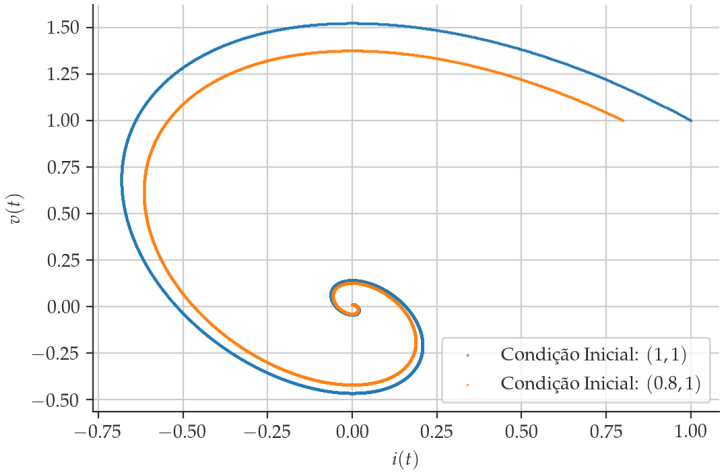

Figura 5: Comparação da evolução temporal no espaço de fase de duas condições iniciais próximas no atrator do circuito RLC

Para atratores usuais, a diferença entre as trajetórias diminuirá progressivamente (no caso do ponto-fixo), se manterá constante (no caso do ciclo-limite), ou aumentará exponencialmente para o infinito, apresentando uma divergência total.

A característica que torna possível que um atrator estranho apresente comportamentos distintos para condições iniciais próximas, sem divergirem para o infinito, é a dimensão fractal nas trajetórias do espaço de estado, conforme detalhado na Seção 2.5.1. Assim, as sucessivas contrações e expansões do espaço de fase, o que forma uma estrutura fractal, fará com que, por menor que seja a diferença entre as condições iniciais, como há uma infinidade de estados possíveis para a evolução dos mesmos,

após algumas iterações do sistema, observaremos que as trajetórias seguiram caminhos distintos entre os estados emaranhados na dimensão fractal.

Além disso, perceba também que as dimensões fractais permitem que haja uma divergência que aumenta exponencialmente, sem divergir para o infinito (o que não ocorre em certos atratores de sistemas que também apresentam divergência). Esse fenômeno é devido ao fato de que as sucessivas expansões e contrações (também chamadas de processo de dobra [4, 7]) estão contidas em um espaço limitado, devido às dimensões fracionárias (a Seção 2.5.3 apresenta essas características de um ponto de vista da teoria da informação).

Interessantemente, a obrigatoriedade de não haver interseções nas trajetórias no espaço de fases leva à seguinte restrição: sistemas dinâmicos a tempo contínuo devem ter, no mínimo, 3 dimensões para poderem exibir comportamento caótico, como demonstrado em [15]. Em contrapartida, sistemas dinâmicos discretos com menos dimensões, como o mapa de Hénon (de duas dimensões), já podem ser caóticos.

Considere o clássico sistema de Lorenz [16], definido pelas equações diferenciais indicadas abaixo:

$$\frac{dx}{dt} = -\sigma \cdot (x - y) \tag{19a}$$

$$\frac{dy}{dt} = x \cdot (\rho - z) - y \tag{19b}$$

$$\frac{dz}{dt} = x \cdot y - \beta \cdot z \tag{19c}$$

A Figura 6 mostra a evolução temporal do sistema de Lorenz no espaço de fase (xyz) a partir de três condições iniciais próximas.

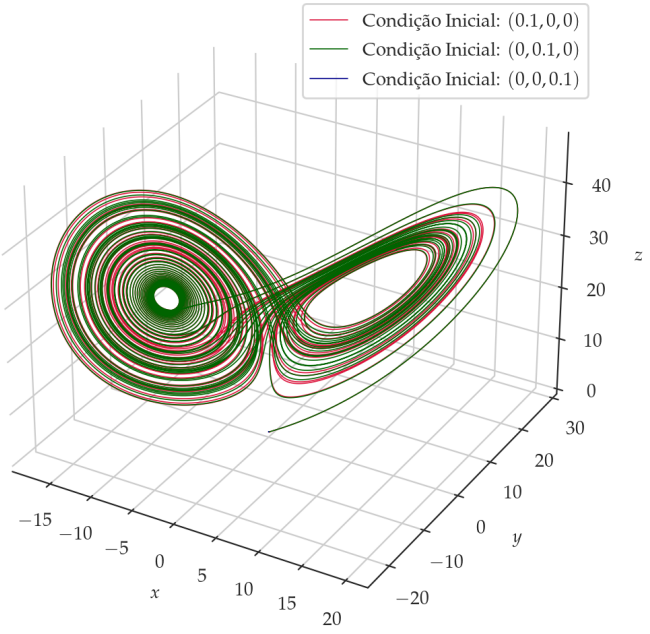

Figura 6: Comparação da evolução temporal no espaço de fase de três condições iniciais próximas para o sistema de Lorenz

A dependência sensitiva em relação às condições iniciais é de tamanha importância para o estudo de sistemas caóticos, que Edward Lorenz, o responsável pelo atrator de Lorenz visto acima, enunciou a seguinte frase na palestra em que pela primeira vez foi apresentado o termo 'efeito borboleta' [10]:

"Uma borboleta batendo as asas no Brasil pode causar um tornado no Texas?"

Por fim, como foi mencionado na Seção 2.3, os atratores estranhos possuem uma propriedade interessante com relação aos seus expoentes de Lyapunov. Como foi dito anteriormente na apresentação dos atratores estranhos, as trajetórias no espaço de fase de um sistema caótico possuem uma divergência que é contida em uma região do espaço, sem irem para o infinito (processo de dobra).

Logo, deve haver pelo menos um expoente de Lyapunov positivo. Porém, sistemas caóticos também são dissipativos, ou seja, é válida a relação:

$$\sum_{i=1}^{m} \lambda_i < 0, \tag{20}$$

e, através dessa relação, pode-se provar o fato de que sistemas caóticos em tempo contínuo necessitam de pelo menos 3 dimensões no espaço de fase.

Para entender por que essa condição é válida, imagine que seja possível obter um atrator estranho em duas dimensões ($m = 2$). Logo, para haver a DSCI, um dos expoentes de Lyapunov é positivo. Porém, uma condição que é válida para toda trajetória no espaço de fase é que, ao longo da direção paralela ao fluxo, o expoente associado é nulo. Então, teríamos um expoente de Lyapunov positivo e outro nulo, o que faria com que $\sum_{i=1}^{2} \lambda_i > 0$, indicando que o elemento de volume divergiria para o infinito, o que não é o caso dos atratores estranhos [4].

Vale mencionar que, apesar dos sistemas caóticos discretos não seguirem essa condição, pode-se mostrar que os mapas (sejam estes uni ou bidimensionais) de seus espaços de estado são amostragens de uma trajetória em três ou mais dimensões, estas sim cumprindo a condição apresentada [4].

2.5.3 Produção de informação em uma dinâmica caótica

Assim como foi apresentado na seção 2.4, sistemas dinâmicos podem ser estudados analisando a produção de informação através da medida de entropia de Kolmogorov-Sinai.

Para tal, considere as relações em (16), que interligam os expoentes de Lyapunov com a produção de informação de um sistema. Consequentemente, quando há pelo menos um expoente de Lyapunov positivo num sistema dinâmico (DSCI), há produção de informação por parte do sistema. Assim, um observador que em $t = 0$ tinha uma certa informação sobre o sistema, à medida que o tempo passa sua informação diminui [4].

Porém, como a relação em (16b) também é cumprida, a produção segue essa desigualdade. Ou seja, apesar do sistema produzir informação com o passar do tempo, esta não diverge exponencialmente para o infinito (o que caracterizaria um processo estocástico).

Para facilitar a visualização da diferença entre os expoentes de Lyapunov, dimensionalidade e entropia de Kolmogorov-Sinai para os casos de sistemas dinâmicos regulares, sistemas caóticos e sistemas de processos estocásticos, a tabela abaixo sintetiza as informações apresentadas nas últimas seções [4]:

Processo	**Expoentes de Lyapunov**	**Entropia de Kolmogorov-Sinai**	**Dimensão da dinâmica assintótica** $(t \to \infty)$
Regular	$\nexists\, \lambda_i > 0$	$K = 0$	$D < m,\ D \in \mathbb{N}$
Caótico	$\exists\, \lambda_i > 0$	$0 < K \leq \sum_{\lambda_i \geq 0} \lambda_i$	$D < m,\ D \in \mathbb{R}$ Sistemas contínuos: $m \geq 3$
Estocástico	—	$K \to \infty$	$D = m$

Tabela 1: Comparação dos Expoentes de Lyapunov, da Entropia de Kolmogorov-Sinai e da Dimensionalidade para sistemas dinâmicos regulares, caóticos e estocásticos

2.5.4 A série temporal e o espectro de potências de um sistema caótico

Outra abordagem possível para o estudo de sistemas dinâmicos caóticos envolve a análise de suas séries temporais.

Para isso, uma ferramenta que permite obter conclusões interessantes, podendo indicar a rápida perda de informação por parte do observador, mencionada na seção anterior, é a análise da autocorrelação e da autocorrelação parcial de uma série temporal.

Através de [4], pode-se definir a função de autocorrelação ϕ_m como a média do produto dos valores de um sinal $x(t)$ em instantes de tempo espaçados de Δt, indicando por quanto tempo o valor do sinal no instante t depende de seus valores prévios e medindo o grau de semelhança existente no sinal à medida que o tempo passa:

$$\phi_m = \frac{1}{N} \sum_{j=1}^{N} x_j x_{j-m} \tag{21}$$

Similarmente, a autocorrelação parcial encontra a correlação dos valores da série x_j e x_{j-m} após o ajuste para remover o efeito da presença dos termos intermediários x_{j-1}, x_{j-2}, ..., x_{j-m+1}.

A Figura 7 exibe a série temporal, a autocorrelação e a autocorrelação parcial para um atraso K na série temporal discreta do mapa de Hénon, mencionado anteriormente, associada à variável x.

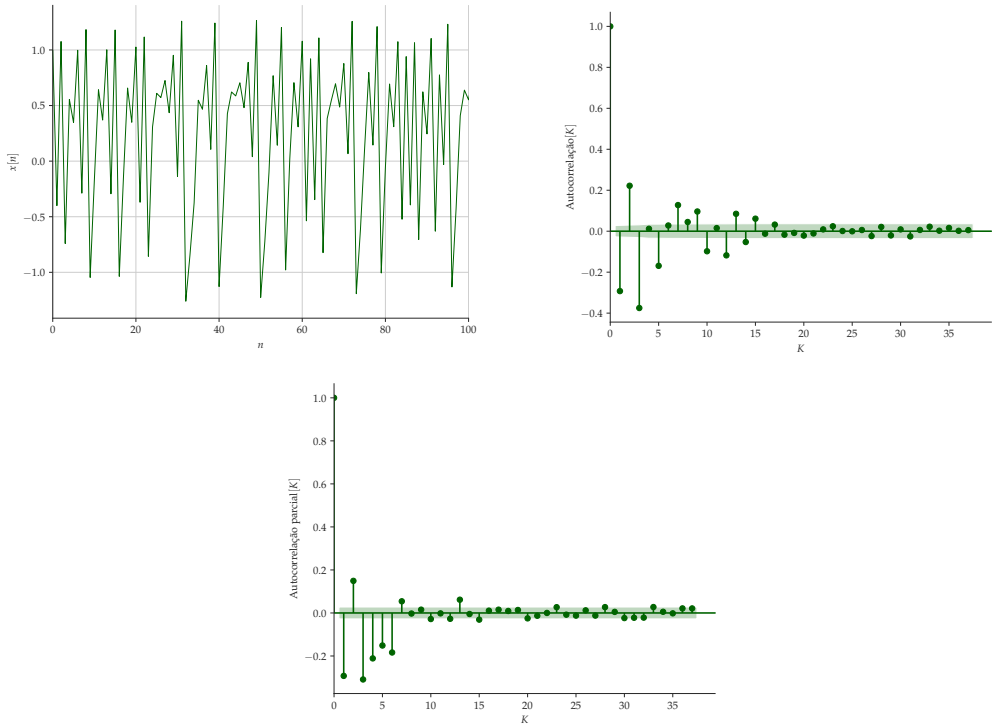

Figura 7: Série temporal, autocorrelação, e autocorrelação parcial para o mapa de Hénon em x

Além do fato que a série temporal é extremamente errática, perceba também que a autocorrelação parcial decai rapidamente, sendo que, a partir de $K = 8$, não há mais uma presença significativa (evidenciado pela faixa verde na figura). Esse fato é importante, pois nos basearemos nestes valores de correlação para estabelecer quantas amostras passadas devem ser fornecidas na entrada dos preditores (falaremos mais sobre isso a partir da Seção 3).

Por fim, o espectro de potências $P(\Omega) = |X(e^{j\Omega})|^2$ do sinal de uma série temporal originada por um sistema caótico também possui propriedades interessantes. A Figura 8 mostra esse espectro para a série temporal do mapa de Hénon exibida acima:

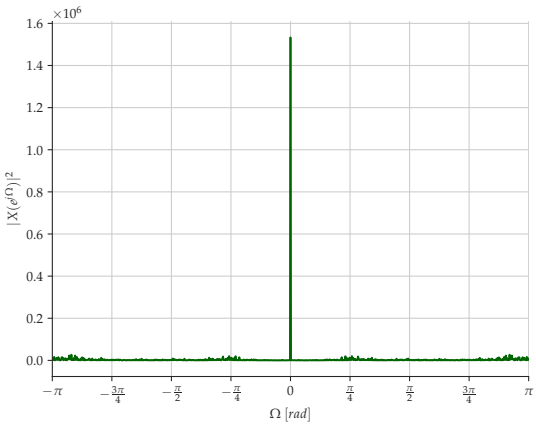

Figura 8: Espectro de potências para a série temporal do mapa de Hénon em x

Analisando o espectro, é possível perceber que ele é bem similar a um espectro de uma série temporal de um processo estocástico, sendo contínuo e de faixa larga. Para descobrir se uma série temporal tem origem em um processo caótico ou estocástico, normalmente são utilizadas técnicas de reconstrução dos atratores desses sistemas [17], inclusive algumas mais recentes utilizando ferramentas de *machine learning* [18].

2.5.5 Características gerais de um sistema com dinâmica caótica

Depois de uma exposição de alguns conceitos fundamentais de sistemas dinâmicos, além de alguns pontos sobre dinâmica caótica, podemos sintetizar o que foi apresentado nas propriedades fundamentais do caos determinístico em sistemas dinâmicos [7]:

1. Forte sensibilidade com respeito às condições iniciais (DSCI);
2. A evolução temporal das variáveis de estado é rápida e tem uma aparência errática;
3. Um sinal originado por um sistema caótico tem espectro de potências contínuo e de faixa larga;
4. Há uma produção de informação por parte do sistema;
5. Presença de atratores estranhos.

Logo, um sistema dinâmico que possui essas propriedades é um sistema caótico.

Nossa análise será sobre a predição das séries temporais originadas por esses sistemas. Por tudo que foi dito até aqui, é esperado que não seja uma tarefa fácil estimar seus valores futuros.

Por causa disso, o estudo se direcionará à aplicabilidade das redes neurais à predição, ou seja, utilizando ferramentas que também são não-lineares, assim como os sistemas em estudo. Logo, na seção a seguir serão discutidos algumas estruturas de redes neurais artificiais, além de uma breve exposição sobre modelos clássicos lineares.

3 Modelos de Predição

3.1 Modelos Lineares

Apesar desta pesquisa se dedicar à aplicação de modelos de predição não-lineares (mais especificamente, algumas opções de redes neurais artificiais), é pertinente darmos uma introdução ao assunto através de modelos lineares para essa aplicação, já aproveitando o momento para apresentarmos alguns conceitos básicos de predição.

3.1.1 Modelo Autorregressivo (AR)

No modelo autorregressivo (AR, do inglês *autoregressive*) o valor da série para um instante de tempo n, denotado por $x(n)$, é dado pela combinação linear dos valores passados a partir do instante $n - L - (K - 1)$ até o instante $n - L$, onde L é o passo de predição (quantos instantes de tempo à frente pretende-se predizer o valor da série) e K é a ordem do modelo.

Portanto, podemos dizer que, no modelo AR o valor da série temporal num instante n é dado por [19]:

$$x(n) = a_1 \cdot x(n - L) + a_2 \cdot x(n - L - 1) + ... + a_K \cdot x(n - L - (K - 1)) + \eta(n) \tag{22}$$

onde a_k, $k = 1, 2, ..., K$ são os coeficientes que ponderam as amostras nos instantes passados e $\eta(n)$ é o erro instantâneo do modelo preditor. Esse erro instantâneo é um ruído branco (do inglês, *white noise*), possuindo média nula e variância σ_n^2 constante [1].

É interessante mencionar que, se considerarmos $L = 1$, ou seja, se estivermos predizendo o valor da série num instante seguinte ao atual, podemos dizer que:

$$\sum_{k=0}^{K} w_k \cdot x(n - k) = \eta(n) \tag{23}$$

sendo $w_0 = 1$ e $w_k = -a_k$ para $1 \leq k \leq K$.

Perceba que o lado esquerdo de (23) é uma soma de convolução em tempo discreto. Portanto, podemos interpretar o modelo AR como um sistema linear e invariante com o tempo (LIT) [19].

3.1.2 Modelo Autorregressivo e de Médias Móveis (ARMA)

O modelo ARMA (do inglês, *auto-regressive moving-average*) incorpora à construção de $x(n)$ valores passados do ruído branco (ou do erro de predição), de forma que [1]:

$$x(n) = \sum_{k=1}^{K} a_k \cdot x(n - L - (k - 1)) + \sum_{k=1}^{M} b_k \cdot \eta(n - L - (k - 1)) \tag{24}$$

Dado um conjunto de observações temporais, a construção de um modelo ARMA que aproxime tal conjunto requer o uso de métodos iterativos e/ou heurísticos de otimização. Isso é devido ao fato de que não há soluções em forma fechada para obter os coeficientes b_k. Além disso, é válido mencionar que durante a otimização desses parâmetros, devemos nos atentar à estabilidade desse sistema, afinal, os erros podem se acumular, levando a uma divergência na saída do preditor [1].

Apesar dos modelos lineares ainda serem bastante utilizados para a predição de séries temporais, para determinadas situações a regra linear aplicada pelos modelos

AR e ARMA não é suficiente para realizar uma predição com um erro aceitável em sistemas mais complexos.

Devido a isso, optamos por direcionar o estudo para modelos de predição não-lineares baseados em redes neurais artificiais. Vejamos, então, como são estes preditores.

3.2 Modelos Não-lineares

Os modelos não-lineares estudados foram as famosas redes neurais artificiais.

As redes neurais artificiais são ferramentas computacionais cujas estruturas são inspiradas no funcionamento das redes neurais biológicas presentes em cérebros de animais desenvolvidos, em especial do ser humano. Podemos interpretar um neurônio (tanto biológico, quanto artificial) como uma unidade de processamento de informação [20].

Analogamente, uma rede neural artificial é uma estrutura formada por vários neurônios artificiais interconectados, a qual é capaz de processar estímulos (sinais) de entrada e de produzir respostas conforme a tarefa desejada. Existem alguns modelos matemáticos para o neurônio artificial, sendo o *perceptron* um dos mais usuais (vide Seção 3.2.1). Além disso, os neurônios podem ser organizados de diferentes maneiras para construir a arquitetura (ou topologia) da rede neural, a qual é tipicamente estruturada em camadas. Por fim, os neurônios artificiais podem exibir uma estrutura interna que varia de acordo com a arquitetura desejada para a aplicação, como observaremos na Seção 3.2.2, onde são discutidos alguns exemplos de redes recorrentes.

3.2.1 Redes *Multilayer Perceptron* (MLP)

Um dos modelos mais utilizados para representar um neurônio artificial, o *Perceptron* [21], é apresentado na Figura 9.

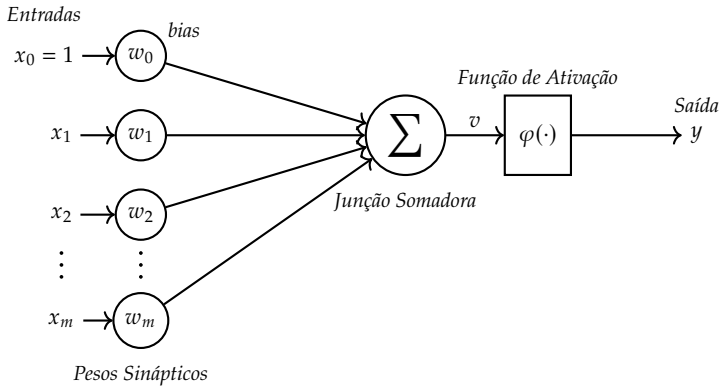

Figura 9: Modelo *Perceptron* para o neurônio artificial

Em termos matemáticos, a saída do neurônio pode ser escrita como:

$$y = \varphi(v) = \varphi\Big(\sum_{i=1}^{m} w_i x_i + w_0\Big) = \varphi\Big(\sum_{i=0}^{m} w_i x_i\Big) = \varphi(\mathbf{w}^T \cdot \mathbf{x}), \tag{25}$$

onde $\mathbf{w}$ é o vetor que contém os coeficientes, denominados de pesos sinápticos, que ponderam as entradas do neurônio.

A escolha da função de ativação $\varphi(\cdot)$ varia de acordo com a aplicação desejada. Ela pode ser desde uma função de *Heaviside*, a puramente linear $\varphi(x) = x$, ou até mesmo a tangente hiperbólica, a função logística ou outras funções não-lineares para mapeamentos mais complexos [6]. Na figura 10, vemos alguns exemplos de funções de ativação comumente utilizadas nos neurônios *Perceptron*, assim como as suas derivadas.

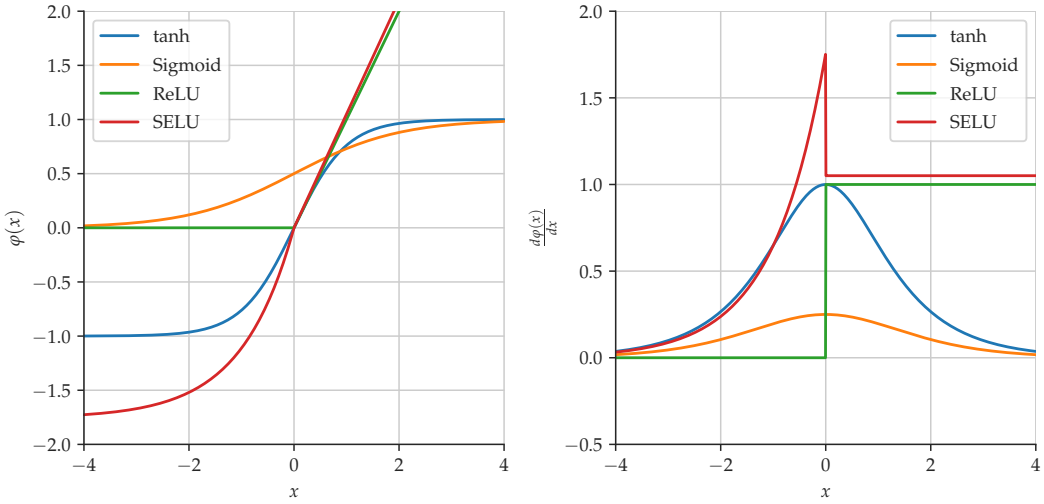

Figura 10: À esquerda, algumas funções de ativação comuns em tarefas de predição e regressão para o neurônio *Perceptron* e, à direita, suas derivadas

É interessante mencionar o fato de que, assim como podemos selecionar uma gama de funções não-lineares para a ativação do neurônio, de forma a realizar transformações não-lineares na entrada, também é possível utilizar funções de ativação lineares, ou seja, realizar transformações lineares à semelhança dos modelos clássicos de predição. Nesse caso, a falta da não-linearidade tornaria muito difícil ou até mesmo impossibilitaria que sistemas mais complexos fossem descritos de forma aceitável por redes neurais artificiais desse tipo [22].

Tipicamente, uma rede neural MLP é composta por um número arbitrário N_L de camadas com n neurônios do tipo *Perceptron*, com a característica de que as saídas dos neurônios da l-ésima camada são propagadas para a frente, servindo como as entradas de todos os neurônios da camada seguinte $(l + 1)$. Esse processo é conhecido como *feedforward*. Por isso, este tipo de rede é conhecida como totalmente conectada (ou densa) e pertence à categoria de modelos *feedforward*. A Figura 11 apresenta a estrutura típica das redes MLP.

Pela figura vemos que, além das N_L camadas intermediárias com neurônios *Percep-*

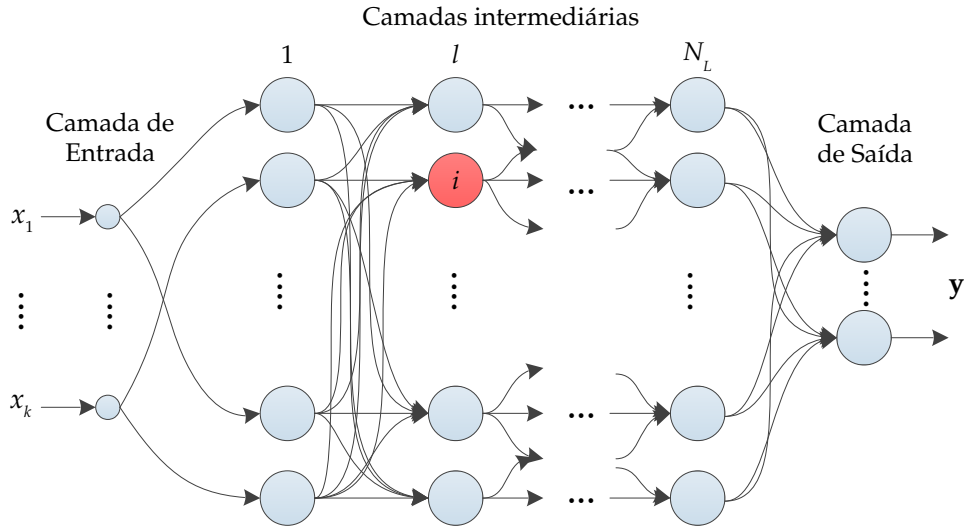

Figura 11: Estrutura típica de uma rede MLP (figura adaptada de [23])

tron, a estrutura das redes MLP contém uma camada de entrada, que possui ativação linear e apenas passa o atributo de entrada relacionado ao neurônio específico ($x_1, x_2, ..., x_k$) para a primeira camada intermediária, e uma camada de saída que gera as respostas da rede neural para um vetor de entrada. A função de ativação da camada de saída e o número de neurônios presentes nela dependem da tarefa a ser realizada pela MLP. Por exemplo, em aplicações de regressão é comum o uso de um ou mais neurônios (dependendo se a saída do preditor será um vetor com um ou mais instantes de tempo) com função de ativação linear. Já em aplicações de classificação, a função de ativação pode variar entre a função logística (utilizada no caso binário em que queremos saber se uma entrada pertence ou não a uma classe) e a função *softmax* (que gera uma saída para cada classe, indicando a probabilidade de uma entrada pertencer à classe em específico) e, novamente, o número de neurônios de saída também varia com o tipo da classificação.

Com base em (25), é possível representar a saída do i-ésimo neurônio da l-ésima camada intermediária da seguinte forma:

$$y_i^l = \varphi^l\Big(\sum_{j=1}^{n_{l-1}} w_{ij}^l y_j^{l-1} + w_{i0}^l\Big), \tag{26}$$

onde w_{ij}^l representa o peso sináptico da conexão que liga o j-ésimo neurônio da camada $l-1$ ao i-ésimo neurônio da camada l, sendo que na primeira camada intermediária os sinais de entrada são os atributos do vetor de entrada, ou seja, $y_j^0 = x_j$ com $j = 1, ..., k$ [23]. O parâmetro n_{l-1} denota o número de neurônios na camada $(l-1)$.

O processo de treinamento de uma rede neural artificial normalmente é realizado com sequências de vetores de entrada $\mathbf{x}$, chamadas de *mini-batch*, e chamamos um período de treinamento de época (do inglês *epoch*) [6]. No treinamento, os pesos sinápticos $\mathbf{w}$ são ajustados em um processo iterativo de forma a minimizar uma função custo $J(\mathbf{w})$, a qual exprime uma medida do erro entre as saídas geradas pela rede e as saídas desejadas (vale mencionar que $\mathbf{w}$ é um vetor com todos os parâmetros da rede). No caso de um problema de regressão ou predição, é comum que a função custo

a ser minimizada seja o Erro Quadrático Médio (MSE, do inglês *Mean Squared Error*). De qualquer forma, o treinamento de uma rede neural dá origem a um problema de otimização não-linear irrestrita [20].

Para isso, é frequente o uso de algoritmos de otimização baseados em derivadas da função custo $J(\mathbf{w})$, como o método do gradiente descendente estocástico (SGD, do inglês *stochastic gradient descent*), o método de Nesterov (NAG, do inglês *Nesterov Accelerated Gradient*) e o algoritmo Adam (*Adaptive Moment Estimation*) [6]. O famoso algoritmo de retropropagação (*backpropagation*) do erro, cuja representação matemática também varia de acordo com o tipo de otimização, é empregado para viabilizar o cálculo das derivadas com relação aos pesos sinápticos dos neurônios situados nas camadas internas da rede, de forma que, para cada *mini-batch*, seja possível atualizar todos os pesos de todos os neurônios presentes em todas as camadas.

Nesse caso, é comum dividirmos os métodos de otimização entre métodos de primeira ordem e de segunda ordem. Os métodos de primeira ordem utilizam as derivadas de primeira ordem da função custo, geralmente representadas com o uso do vetor gradiente:

$$\nabla J(\mathbf{w}) = \left[\frac{\partial J(\mathbf{w})}{\partial w_1} \ \ldots \ \frac{\partial J(\mathbf{w})}{\partial w_n} \right]^T \tag{27}$$

Assim, ao caminharmos na direção contrária à apontada pelo vetor $\nabla J(\mathbf{w})$ obtemos, de forma iterativa, a minimização desejada. Logo, a regra de atualização dos pesos pode ser dada pela forma básica:

$$\mathbf{w}[k+1] \leftarrow \mathbf{w}[k] - \eta \nabla J\big[\mathbf{w}[k]\big] \tag{28}$$

Os algoritmos de otimização mencionados (SGD, Adam, NAG) utilizam variações da regra de atualização de $\mathbf{w}$ indicada em (28).

Para esta pesquisa, optamos por utilizar o algoritmo Nadam (*Nesterov Adaptive Moment Estimation*) [24]. O Nadam, como o nome já sugere, incorpora elementos dos algoritmos NAG e Adam, de forma a ter um passo de atualização adaptativo e com acúmulo dos gradientes mais recentes (semelhante à ideia de momento linear de uma partícula), com a utilização do "truque de Nesterov", desenvolvido pelo matemático russo Yurii Nesterov [25], onde o gradiente não é calculado sobre o ponto atual indicado pelo vetor $\mathbf{w}$, mas sim sobre um ponto levemente à frente, na mesma direção do momento. O conjunto de equações que descreve o algoritmo Nadam é dado por:

$$\mathbf{w}' = \mathbf{w} + \beta_3 \mathbf{m} \tag{29a}$$

$$\mathbf{m} \leftarrow \beta_1 \mathbf{m} - (1-\beta_1) \nabla_{\mathbf{w}} E(\mathbf{w}') \tag{29b}$$

$$\mathbf{s} \leftarrow \beta_2 \mathbf{s} - (1-\beta_2) \nabla_{\mathbf{w}} E(\mathbf{w}') \otimes \nabla_{\mathbf{w}} E(\mathbf{w}') \tag{29c}$$

$$\hat{\mathbf{m}} \leftarrow \frac{\mathbf{m}}{1-\beta_1} \tag{29d}$$

$$\hat{\mathbf{s}} \leftarrow \frac{\mathbf{s}}{1-\beta_2} \tag{29e}$$

$$\mathbf{w} \leftarrow \mathbf{w} + \eta\hat{\mathbf{m}} \oslash \sqrt{\hat{\mathbf{s}} + \epsilon} \tag{29f}$$

A incorporação dos gradientes passados no cálculo do próximo vetor $\mathbf{w}$ ocorre através da equação (29b), que define o vetor de momento $\mathbf{m}$. Assim, utilizando a analogia com o momento linear de uma partícula, o gradiente passa ter a ideia de uma força de aceleração que aumenta o valor do momento do vetor $\mathbf{w}$ caso este esteja indo para um ponto ótimo de mínimo local, escapando mais rapidamente de regiões onde não há minimização significativa da função custo.

O hiperparâmetro β_1 presente em (29b) serve tanto para evitar que $\mathbf{m}$ aumente de forma descontrolada, como também para fazer com que, através do termo $(1-\beta_1)$, surja uma média exponencialmente decrescente de valores anteriores. Normalmente, β_1 é inicializado em 0.9.

Já o vetor $\mathbf{s}$, definido em (29c), serve para evitar que o algoritmo de otimização caminhe com uma direção que não aponte para o ponto ótimo mencionado anteriormente. Além disso, para evitar que esse ajuste na direção cause uma perda significativa na rapidez de convergência do algoritmo, utiliza-se os termos β_2 e $(1-\beta_2)$, ou seja, um decaimento exponencial nos valores passados de $\mathbf{s}$. Nesse caso, o hiperparâmetro β_2 é inicializado em 0.999 e ϵ em 10^{-7} (para evitar uma divisão por zero).

Por fim, as equações (29d) e (29e) servem para corrigir um possível viés dos vetores $\mathbf{m}$ e $\mathbf{s}$ devido à estratégia típica de inicialização (a saber, para valores próximos a 0).

É importante mencionar que os métodos de otimização aqui mencionados são métodos de busca local, ou seja, têm convergência esperada para um mínimo local, que não necessariamente é o mínimo global da função custo para a aplicação. Dizemos também que esses pontos mínimos possuem "bacias de atração"que, para valores adequados da taxa de aprendizado η, atraem o vetor de parâmetros $\mathbf{w}$ [5].

Um fenômeno possivelmente presente durante o treinamento da rede neural é o desaparecimento dos gradientes (do inglês *vanishing gradients problem*) nas camadas inferiores da rede, ou o aumento desenfreado desses gradientes (do inglês *exploding gradients problem*) nas camadas citadas, que é mais comum para redes recorrentes. Para evitar que isso ocorra, normalmente são utilizadas camadas que realizam uma normalização do *batch* de vetores de entrada (BN, do inglês *Batch Normalization*), sendo posicionadas antes ou depois das camadas intermediárias da rede [6].

De forma resumida, o que a BN faz é remover a média e padronizar os desvios de todas as variáveis em sua presente, conforme a informação disponível no *mini-batch* atual, para, depois, redefinir tais medidas com parâmetros que devem ser determinados pelo treinamento. Ou seja, a camada BN tenta identificar a forma de melhor remodelar as variáveis de entrada em termos de suas médias e desvios padrão. A transformação aplicada pela BN pode ser representada pelas seguintes equações:

$$\boldsymbol{\mu}_B = \frac{1}{d}\sum_{i=1}^{d} \mathbf{x}_i \tag{30a}$$

$$\sigma_B^2 = \frac{1}{d}\sum_{i=1}^{d}(\mathbf{x}_i - \boldsymbol{\mu}_B)^2 \tag{30b}$$

$$\hat{\mathbf{x}}_i = \frac{\mathbf{x}_i - \boldsymbol{\mu}_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{30c}$$

$$\mathbf{z}_i = \boldsymbol{\gamma} \otimes \hat{\mathbf{x}}_i + \boldsymbol{\beta} \tag{30d}$$

sendo $\boldsymbol{\mu}_B$ e $\boldsymbol{\sigma}_B$ os vetores com as médias e desvios-padrão para cada atributo de entrada do atual *mini-batch* (este contendo d vetores $\mathbf{x}$), $\hat{\mathbf{x}}_i$ é o i-ésimo vetor de entrada centrado em zero e normalizado, $\boldsymbol{\gamma}$ é o vetor que pondera cada um dos atributos de $\hat{\mathbf{x}}_i$, $\boldsymbol{\beta}$ tem um termo de *offset* para cada atributo normalizado, ϵ novamente é um termo pequeno (geralmente na ordem de 10^{-5}) para evitar uma divisão por zero e, por fim, $\mathbf{z}_i$ é a saída da camada de *batch normalization*.

Os hiperparâmetros $\boldsymbol{\beta}$ e $\boldsymbol{\gamma}$ controlam o deslocamento e o escalonamento, respectivamente, de cada atributo para um vetor de entrada. Essas variáveis são parametrizadas durante o treinamento da rede neural de forma bem eficiente dependendo da quantidade e do tamanho dos *mini-batches*.

Também vale mencionar que, para o conjunto de dados de validação, ou seja, o conjunto de dados que, após o ajuste dos pesos $\mathbf{w}$ no conjunto de treinamento, é verificado se a rede neural está generalizando os resultados, evitando o sobreajuste (do inglês *overfitting*), são calculados outros vetores de valores de média ($\boldsymbol{\mu}'_B$) e desvios-padrão ($\boldsymbol{\sigma}'_B$), a partir de médias móveis com decaimento exponencial. Assim, o processo descrito anteriormente também é realizado sobre esses dados.

Por fim, o grande apelo das redes *Multilayer Perceptron* é que elas possuem capacidade de aproximação universal, ou seja, são capazes de aproximar qualquer mapeamento contínuo num domínio compacto com um nível de erro arbitrariamente pequeno. Até mesmo uma MLP com uma única camada intermediária e camada de saída linear já possui esta capacidade [26, 22]. Infelizmente, esse teorema não indica a quantidade de neurônios necessária na(s) camada(s) intermediária(s), muito menos um método para ajustar o vetor $\mathbf{w}$ da rede para garantir a solução ótima.

Além disso, para o propósito deste trabalho de pesquisa, que envolve sistemas dinâmicos, ter acesso a um histórico e a uma memorização dos padrões anteriores de entrada, na teoria, aumentaria a precisão da predição das séries temporais. Mesmo que as redes MLP tenham capacidade de aproximação universal, é possível aumentar a flexibilidade desse processo caso fossem utilizadas estruturas recorrentes [2, 23].

Logo, incluímos na análise alguns modelos de redes neurais recorrentes. Discutiremos mais sobre a estrutura e o funcionamento delas na próxima seção.

3.2.2 Redes Neurais Recorrentes (RNN)

Diferentemente das redes MLP que são *feedforward*, ou seja, que não reutilizam a informação processada dos padrões anteriores para gerar a próxima saída, a ideia central

das redes neurais recorrentes (RNN, do inglês *Recurrent Neural Networks*) é que elas têm estruturas computacionais que podem armazenar os estados anteriores dos neurônios, podendo também possuir portas não-lineares que regulam o fluxo de informação de entrada e de saída da célula computacional [20]. Uma representação possível de uma célula básica de uma rede recorrente pode ser vista na Figura 12 (esquerda).

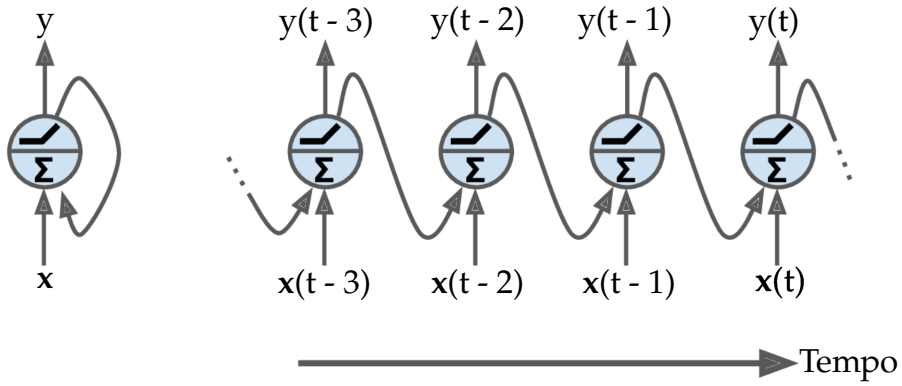

Figura 12: À esquerda, célula básica da rede recorrente e, à direita, sua representação através do tempo (figura adaptada de [6])

Note que a saída é realimentada (com um atraso temporal) para a entrada do próprio neurônio. Assim, a célula recorrente recebe tanto o vetor de entradas $\mathbf{x}(t)$ assim como a saída no instante anterior $y(t-1)$ que, devido à relação de recorrência, reflete implicitamente as saídas em todos os instantes anteriores $y(t-k)$ com $1 < k \leq t$ (considerando $y(0)$). Dessa forma, como pode ser visto na Figura 12 (direita), é possível representar essa rede ao longo do tempo, através de um processo chamado desenrolamento da rede no tempo (do inglês *unrolling the network through time*) [6].

Uma camada recorrente é formada de maneira similar à célula, com a diferença que cada neurônio recebe, além do vetor de entradas $\mathbf{x}(t)$, um vetor de saídas $\mathbf{y}(t-1)$, que contém as saídas de todos as unidades recorrentes da camada em instantes anteriores, conforme a Figura 13 indica.

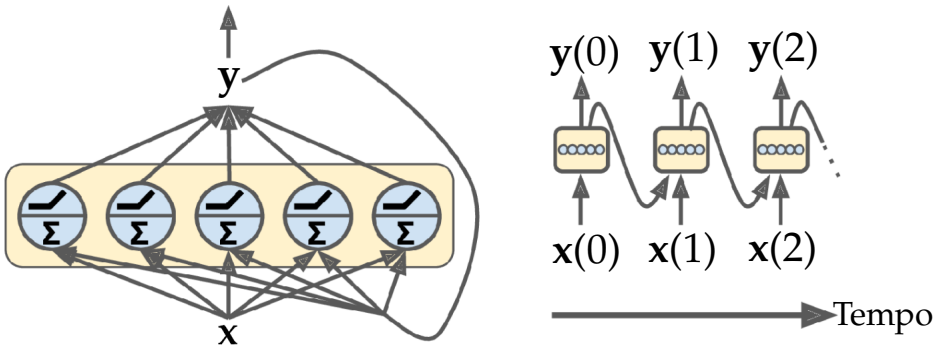

Figura 13: À esquerda, camada da rede recorrente e, à direita, sua representação através do tempo (figura adaptada de [6])

Dessa forma, cada célula recorrente possui dois vetores de pesos, $\mathbf{w}_x$ e $\mathbf{w}_y$. O

primeiro pondera o vetor de entradas $\mathbf{x}(t)$, de forma similar à apresentada anteriormente na MLP, enquanto o vetor $\mathbf{w}_y$ pondera o vetor das saídas dos estados passados $\mathbf{y}(t-1)$.

Logo, pode-se formar uma camada recorrente agrupando os vetores de pesos de cada célula nas matrizes $\mathbf{W}_x$ e $\mathbf{W}_y$, sendo que a saída da camada, isto é, $\mathbf{y}(t)$, é dada por:

$$\mathbf{y}(t) = \varphi\left(\mathbf{W}_x\mathbf{x}(t) + \mathbf{W}_y\mathbf{y}(t-1) + \mathbf{b}\right) \tag{31}$$

Assim, devido à presença do vetor $\mathbf{y}(t-1)$ em (31), a saída da camada recorrente no instante t acaba sendo influenciada por todas as entradas anteriores ($\mathbf{x}(0)$, $\mathbf{x}(1)$, ... $\mathbf{x}(t-1)$). Devido a isso, é dito que a rede recorrente possui uma memória dos estados anteriores [6].

Em geral, o estado atual de uma célula é representado através de uma função $\mathbf{h}(t)$, de forma a termos a seguinte relação entre $\mathbf{h}(t)$ e $\mathbf{y}(t)$:

$$\mathbf{h}(t) = f(\mathbf{x}(t), \mathbf{h}(t-1)) \tag{32}$$

$$\mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{h}(t-1)) \tag{33}$$

Nas células mais básicas apresentadas anteriormente, temos que $f(\mathbf{x}(t), \mathbf{h}(t-1)) = g(\mathbf{x}(t), \mathbf{h}(t-1))$, mas esse nem sempre é o caso. Assim, uma representação mais geral de uma célula recorrente pode ser vista na Figura 14.

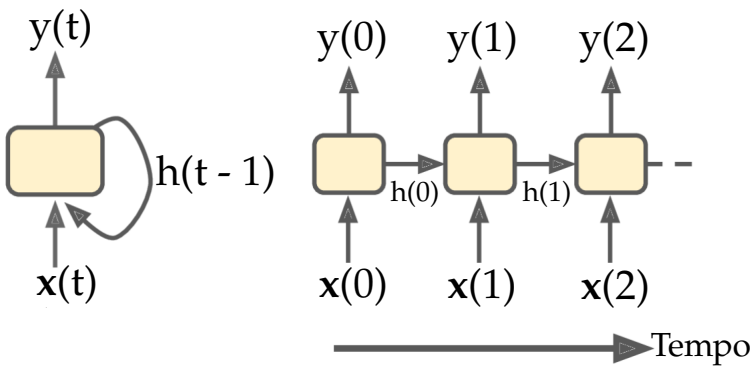

Figura 14: À esquerda, representação geral da célula da rede recorrente e, à direita, sua representação através do tempo (figura adaptada de [6])

A saída de uma RNN varia com a aplicação. Para a predição de séries temporais, utilizamos o arranjo sequência para sequência (do inglês *sequence-to-sequence network*). Neste caso, a rede recebe uma sequência de entradas e gera uma sequência de saídas. Por exemplo, sendo $x(t)$ o valor da série temporal, $y(t)$ é a estimação da rede para o instante $t + L$, sendo L o passo de predição. Logo, fornecendo em $t = 0$ vetores de entrada um a um, as saídas serão as estimações dos valores para os instantes 1 a $N + L$, sendo N o número de amostras da série.

Outro tipo de configuração possível para a predição de séries temporais é o arranjo sequência para vetor (do inglês *sequence-to-vector network*), onde ignoramos todas as saídas com exceção da última. Assim, com esse arranjo é possível predizer os próximos

L valores desconhecidos da série. Essa configuração também é muito utilizada para gerar, com base em uma frase, um *score* para o sentimento que foi transmitido pelo autor [6].

Além disso, também é possível termos um arranjo vetor para sequência (do inglês *vector-to-sequence network*). Esse tipo não é tanto utilizada para a predição de séries temporais, sendo mais utilizada para gerar legendas para imagens.

Por fim, podemos combinar o esquema sequência-vetor seguido do vetor-sequência. Essa configuração é chamada de *Encoder-Decoder*, utilizado para gerar traduções com base em uma frase. O *Encoder* recebe uma sequência de palavras como entrada, gerando um vetor que é decodificado pelo *Decoder* para formar o texto em outro idioma. A Figura 15 exibe as configurações possíveis.

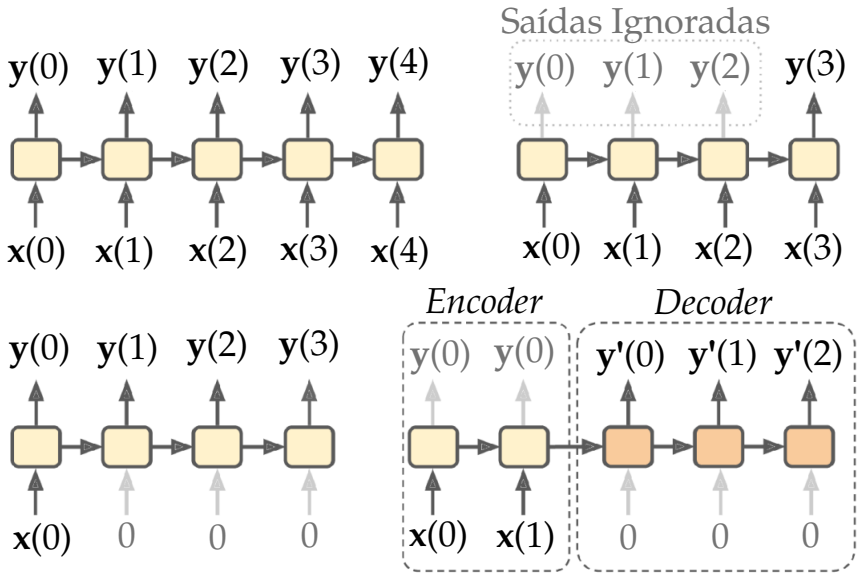

Figura 15: Maneiras de se implementar uma RNN (figura adaptada de [6])

O treinamento de uma rede recorrente também utiliza a técnica de retropropagação do erro, citada anteriormente quando foi discutida a rede MLP. A diferença é que o erro também deve ser retropropagado para k instantes anteriores de tempo. Esta extensão da técnica é chamada de retropropagação do erro através do tempo (BPTT, do inglês *backpropagation through time*).

Para compreender os pontos principais do funcionamento desse algoritmo, convém considerar a rede desdobrada no tempo (Figura 13). Dessa forma, pode-se perceber que uma rede recorrente desdobrada no tempo é bem semelhante a uma rede *feedforward* profunda, em que as várias camadas que representam a rede em instantes de tempo diferente compartilham os mesmos parâmetros.

Nesse caso, a entrada de camada da rede desdobrada corresponde ao vetor de entrada para um determinado instante de tempo. Logo, a camada correspondente ao primeiro instante de tempo recebe a entrada $\mathbf{x}(t-k)$, a camada seguinte a ela recebe a entrada $\mathbf{x}(t-k+1)$, e isso ocorre até a camada mais recente no tempo da rede, que recebe a entrada $\mathbf{x}(t)$. Também é válido mencionar que as saídas de determinados instantes

de tempo podem ser ignoradas ao calcular a função custo J para a rede desdobrada, assim como indica a Figura 16.

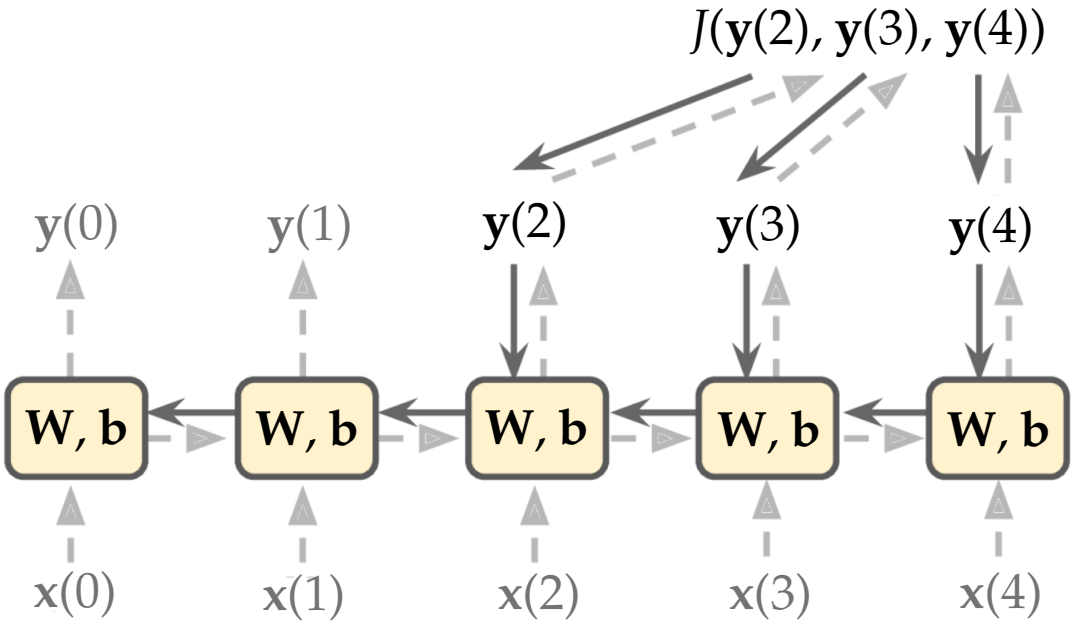

Figura 16: Retropropagação do erro através do tempo (figura adaptada de [6])

Logo, o algoritmo calcula a função custo J, considerando as saídas relevantes, e os gradientes dessa função para os pesos **W** e **b** são propagados em direção aos instantes de tempo anteriores (ignorando os instantes com saídas irrelevantes) e, como esses pesos são os mesmos em todas as camadas da rede desdobrada, eles são ajustados combinando a direção do gradiente calculado em cada instante de tempo. Este processo sintetiza a essência da retropropagação do erro através do tempo.

É importante mencionar que essa técnica também está sujeita a obstáculos semelhantes aos mencionados anteriormente sobre a retropropagação do erro tradicional, como gradientes explosivos e/ou desvanecimento dos mesmos. Além disso, como a rede recorrente é um sistema dinâmico, podem ocorrer problemas com relação à instabilidade do processo de treinamento.

Para mitigar esses problemas, são utilizadas técnicas de *gradient clipping* e normalizações [6], além do uso de funções de ativação baseadas nas funções tangente hiperbólica (tanh) e logística (Sigmoid), devido às saturações das mesmas para entradas muito grandes, ou muito pequenas (Figura 10).

A célula recorrente apresentada aqui é bem básica, aprendendo apenas padrões curtos. Assim, na seção seguinte apresentaremos a rede recorrente LSTM, que possui uma estrutura mais elaborada e é capaz de aprender padrões temporais mais complexos.

3.2.3 Redes *Long Short-term Memory* (LSTM)

Devido às transformações que uma rede recorrente convencional aplica em uma entrada, parte da informação original transmitida é perdida e, depois de várias iterações, não sobra traços das primeiras entradas.

As redes LSTM (do inglês, *Long Short-term Memory*) contornam esse problema inserindo portas dentro da célula recorrente que controlam o fluxo de informação [27].

O neurônio da rede LSTM é bem semelhante ao neurônio básico da rede recorrente mostrado anteriormente. A diferença é a presença de um vetor de longo prazo **c** e um vetor de curto prazo **h**, conforme indicado na Figura 17.

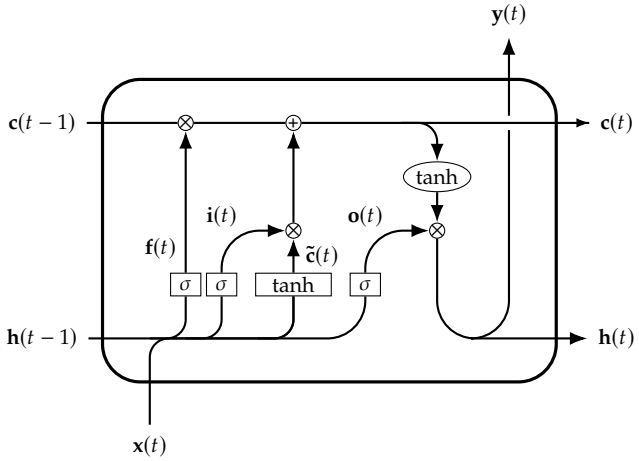

Figura 17: Estrutura interna de uma célula LSTM

Durante o processo de treinamento, a rede aprende o que deve ser armazenado no vetor $\mathbf{c}(t)$ e, do que foi guardado, o que deve ser descartado e o que deve ser levado em conta na hora de estimar a próxima saída.

A primeira operação realizada nesse vetor é a multiplicação elemento a elemento ($\otimes$) com o vetor $\mathbf{f}(t)$. Essa operação também é chamada de porta do esquecimento (do inglês, *forget gate*) pois ela determina quais elementos de $\mathbf{c}(t-1)$ serão descartados, ou seja, quais memórias de instantes passados no longo prazo serão esquecidas.

Após a incorporação de novas informações na operação de soma elemento a elemento ($\oplus$), o vetor $\mathbf{c}(t-1)$ é mandado para a saída da célula e uma cópia sua passa pela função tangente hiperbólica e é filtrado pela porta de saída (do inglês, *output gate*), formando o vetor de curto prazo $\mathbf{h}(t)$, que é também a saída $\mathbf{y}(t)$.

A entrada do neurônio LSTM, além de incluir o vetor $\mathbf{c}(t)$, também inclui os vetores $\mathbf{h}(t-1)$ e $\mathbf{x}(t)$. A partir destas informações, são gerados os vetores $\mathbf{f}(t)$, $\mathbf{i}(t)$, $\tilde{\mathbf{c}}(t)$ e $\mathbf{o}(t)$.

Perceba que o vetor $\tilde{\mathbf{c}}(t)$ é a saída de uma célula recorrente básica, considerando a

tangente hiperbólica como função de ativação. Ou seja, ela é composta pela ponderação entre os estados anteriores de curto prazo $\mathbf{h}(t)$ e a entrada atual $\mathbf{x}(t)$, além do vetor de *bias* $\mathbf{b}(t)$. A LSTM guarda as partes mais relevantes do vetor $\tilde{\mathbf{c}}(t)$ no vetor $\mathbf{c}(t-1)$ através da soma elemento a elemento mencionada anteriormente.

Os outros vetores restantes representam operações de controle do fluxo de informação dentro do neurônio, cada termo variando de 0 a 1 devido à função logística (σ). Assim, ao serem multiplicadas elemento a elemento, controlam quais informações serão eliminadas.

No caso, $\mathbf{f}(t)$ controla quais partes de $\mathbf{c}(t-1)$ serão apagadas, $\mathbf{i}(t)$ controla quais informações de $\tilde{\mathbf{c}}(t)$ serão agregadas ao vetor $\mathbf{c}(t-1)$ e o vetor $\mathbf{o}(t)$ controla quais componentes de $\mathbf{c}(t-1)$ deverão formar a saída $\mathbf{y}(t)$ e o vetor de curto prazo $\mathbf{h}(t)$.

O conjunto de equações abaixo mostra as operações presentes na estrutura interna da célula LSTM, assim como a saída para o instante t, denotada por $\mathbf{y}(t)$:

$$\mathbf{f}(t) = \sigma(\mathbf{W}_f[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_f) \tag{34a}$$

$$\mathbf{i}(t) = \sigma(\mathbf{W}_i[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_i) \tag{34b}$$

$$\tilde{\mathbf{c}}(t) = \tanh(\mathbf{W}_c[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_c) \tag{34c}$$

$$\mathbf{c}(t) = \mathbf{f}(t) \otimes \mathbf{c}(t-1) + \mathbf{i}(t) \otimes \tilde{\mathbf{c}}(t) \tag{34d}$$

$$\mathbf{o}(t) = \sigma(\mathbf{W}_o[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_o) \tag{34e}$$

$$\mathbf{y}(t) = \mathbf{h}(t) = \mathbf{o}(t) \otimes \tanh(\mathbf{c}(t)) \tag{34f}$$

À semelhança das redes recorrentes convencionais, o treinamento de uma LSTM também é realizado através de algoritmos de otimização baseados em derivadas da função custo propagadas ao longo da estrutura e ao longo do tempo, utilizando o algoritmo BPTT.

Resumidamente, as LSTMs manipulam o vetor de longo prazo $\mathbf{c}(t)$, aprendendo durante o treinamento o que deve ser guardado nele, o que deve ser descartado e o que deve ser aproveitado para gerar a saída $\mathbf{y}(t)$ e o vetor de curto prazo $\mathbf{h}(t)$. Dessa forma, podemos dizer que a atualização do vetor de estados $\mathbf{c}(t)$ é feita com o descarte de informações e a incorporação de novidades vindas da entrada.

Depois de uma exposição sobre os modelos preditores que utilizaremos nessa pesquisa, concluiremos este relatório com a indicação dos próximos passos do estudo, já definindo quais são os sistemas caóticos que analisaremos com as redes neurais artificiais citadas.

4 Próximos Passos

Como nessa primeira parte da iniciação o foco foi uma pesquisa bibliográfica dos temas a serem estudados nela, a segunda metade será voltada para a aplicação em si

da predição das séries temporais de sistemas caóticos.

Para os experimentos computacionais, optamos pelas séries temporais associadas ao mapa logístico, descrito pelo cientista Robert May [28], ao mapa de Hénon, apresentado pelo astrônomo e matemático francês Michel Hénon [13], ao sistema de Mackey-Glass, dos cientistas Michael Mackey e Leon Glass [29], e ao clássico sistema de Lorenz, um dos mais incríveis e fundamentais trabalhos de sistemas caóticos, introduzido pelo matemático e meteorologista Edward Norton Lorenz [16]. Este grupo de séries foi selecionado com o intuito de criar cenários diversificados para a análise do comportamento das diferentes redes neurais, considerando sistemas a tempo contínuo (Lorenz e Mackey-Glass) e sistemas a tempo discreto (mapa de Hénon e mapa logístico).

Em seguida, determinaremos aspectos mais fundamentais das redes neurais que serão utilizadas, como, por exemplo, a arquitetura empregada, assim como as métricas para o treinamento e análise. Para essa etapa, também planejamos um estudo das redes GRU (do inglês *Gated Recurrent Unit*) [30] e ESN (do inglês *Echo State Network*) [3], sendo que a última, em outros trabalhos de pesquisa, já indicou um bom desempenho preliminar no contexto de predição de séries temporais originadas por sistemas caóticos [23].

Após isso, faremos a aplicação das redes neurais à predição das séries escolhidas, avaliando a sensibilidade paramétrica de cada estrutura na busca das melhores configurações, a fim de traçar um quadro comparativo entre as técnicas consideradas.

Por fim, compilaremos os resultados no relatório final, de forma a conter uma discussão ampla e representativa dos ensaios realizados e das conclusões obtidas.

Referências

[1] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[2] J. T. Connor, R. D. Martin, and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 240–254, 1994.

[3] H. Jaeger, "Echo state network," *scholarpedia*, vol. 2, no. 9, p. 2330, 2007.

[4] N. Fiedler-Ferrara and C. P. C. do Prado, *Caos: uma introdução*. Editora Blucher, 1994.

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[6] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[7] R. R. de Faissol Attux, "Sobre dinâmica caótica e convergência em algoritmos de equalização autodidata," dissertação (mestrado), Universidade Estadual de

Campinas, Faculdade de Engenharia Elétrica e de Computação, Campinas, SP, 2001.

[8] C. A. Kluever, *Dynamic systems: modeling, simulation, and control*. John Wiley & Sons, 2020.

[9] A. V. Oppenheim, A. S. Willsky, and S. Hamid, *Signals & Systems*. Prentice Hall, 2 ed., 1996.

[10] J. Gleick, *Chaos: The amazing science of the unpredictable*. Vintage Publishing, 1998.

[11] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[12] B. Mandelbrot, "How long is the coast of britain? statistical self-similarity and fractional dimension," *science*, vol. 156, no. 3775, pp. 636–638, 1967.

[13] M. Hénon, "A two-dimensional mapping with a strange attractor," *Communications in Mathematical Physics*, vol. 50, pp. 69–77, feb 1976.

[14] D. Ruelle and F. Takens, "On the nature of turbulence," *Les rencontres physiciens-mathématiciens de Strasbourg-RCP25*, vol. 12, pp. 1–44, 1971.

[15] J. Guckenheimer and P. Holmes, *Nonlinear oscillations, dynamical systems, and bifurcations of vector fields*, vol. 42. Springer Science & Business Media, 2013.

[16] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[17] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, "Geometry from a time series," *Physical review letters*, vol. 45, no. 9, p. 712, 1980.

[18] Z. Lu, B. R. Hunt, and E. Ott, "Attractor reconstruction by machine learning," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 6, p. 061104, 2018.

[19] S. S. Haykin, *Adaptive filter theory*. Pearson Education India, 2008.

[20] S. Haykin, *Neural networks and learning machines, 3/E*. Pearson Education India, 2010.

[21] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[22] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[23] L. Boccato, *Novas propostas e aplicações de redes neurais com estados de eco*. Tese (doutorado), Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação, Campinas, SP, 2013.

[24] T. Dozat, "Incorporating nesterov momentum into adam," 2016.

[25] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate $O\,(1/k^2)$," *Dokl. Akad. Nauk SSSR*, vol. 269, pp. 543–547, 1983.

[26] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[27] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[28] R. M. May, "Simple mathematical models with very complicated dynamics," *Nature*, vol. 261, pp. 459–467, jun 1976.

[29] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, no. 4300, pp. 287–289, 1977.

[30] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.